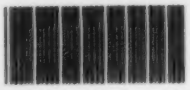
CA1
DTC800-
89
C515

431384

41.D



TP 9990E

CORES: AIR TRAFFIC CONFLICT RESOLUTION SYSTEM

FINAL REPORT

by:

I.G. Smith, C.A. Crabb, G.R. Ram, M.R. Attisha
CompEngServ Ltd.
600 - 265 Carling Avenue
Ottawa, Ontario
K1S 2E1
613-563-1920

October 4, 1989

Library and Information Centre
Centre de documentation
RECEIVED
FEB 27
Transport Canada / Transports Canada

90-04072/

DISCLAIMER: The contents of this report reflect the views of CompEngServ Ltd. and not necessarily the official view or opinions of the Transportation Development Centre of Transport Canada.

LANGUAGE NOTICE: Un sommaire en francais de ce rapport est inclus avant la Table des matieres.

Project Team

Team Member	Title	Role in Project
B. Archie Bowen	President	Technical Advisor
David G. Bowen	Director of Engineering	Project Management
Ian G. Smith	Project Leader	Project Leader
Mark R. Attisha	Project Leader	Software Development
Geetha Ram	Systems Designer	Software Development
Cheryl A. Crabb	Systems Designer	Software Development
Anita Acheson	Senior Programmer	Software Development
James D. Reid	Senior Development Officer	Project Officer
Pierre Sauve	Supervisor Rules and Procedures Development Air Traffic Services	Technical Advisor
Wayne Sherriff	Research Analyst Air Traffic Services Research & Experimentation Centre	Technical Advisor

ACKNOWLEDGEMENTS

The CORES project was funded by the Transportation Development Centre (TDC) of Transport Canada under the direction of Mr. James Reid, Senior Development Officer. Mr. Pierre Sauve of Air Traffic Rules and Procedures Development, Transport Canada and Mr. Wayne Sherriff of the Research and Experimentation Centre, Transport Canada, provided the controller expertise necessary to develop, test, and validate the system.

The direction for extending this work towards both flight planning and tactical radar environment applications was influenced by Mr. Robert Armstrong, Manager of the Research and Experimentation Centre, Mr. Hugh McCallum of the RAMP project office, and Mr. Micha Avni of TDC.

Transport Canada / Transports Canada

PUBLICATION DATA FORM

1. Transport Canada Publication No.	2. Project No.	3. Recipient's Catalogue No.
TP 9990E	7049	

4. Title and Subtitle	5. Publication Date
CORES: Air Traffic Conflict Resolution System Final Report	October 1989
	6. Performing Organization Document No. —
7. Author(s) I.G. Smith, G.R. Ram, M.R. Attisha, C.A. Crabb	8. Transport Canada File No. D 1455-200-1
9. Performing Organization Name and Address CompEngServ Ltd. 265 Carling Avenue, Suite 600 Ottawa, Ontario K1S 2E1	10. DSS File No. 11SD.T8200-7-7562
	11. DSS or Transport Canada Contract No. T8200-7-7562/01-SD
12. Sponsoring Agency Name and Address Transportation Development Centre (TDC) Guy Favreau Complex 200 René Lévesque Blvd. West West Tower, Suite 601 Montreal, Quebec H2Z 1X4	13. Type of Publication and Period Covered Final Report 06/88 – 08/89
	14. Project Officer James D. Reid

15. Supplementary Notes (Funding programs, titles of related publications, etc.)

CORES User's Guide, Testing Document, and Software Design, Vol. 1–5, published under separate cover.

16. Abstract

This document describes a proof-of-concept advisory system for air traffic conflict resolution, known as CORES (COnflict REsolution System). CORES is a stand-alone system that simulates the movement of aircraft, identifies aircraft conflicts, provides resolution alternatives, and allows for the examination of these procedures. The system is based upon the non-radar enroute environment typical of northern Canada. CORES demonstrates the feasibility of an operational conflict resolution advisory system. It provides a vehicle for testing the effectiveness of an automated advisory function under a wide range of circumstances. This document provides an overview of CORES, an evaluation of the system, and recommendations for future work.

17. Key Words	18. Distribution Statement
Air traffic control, conflict resolution, expert systems	Limited number of copies available from the Transportation Development Centre

19. Security Classification (of this publication)	20. Security Classification (of this page)	21. Declassification (date)	22. No. of Pages	23. Price
Unclassified	Unclassified	—	49	—

TDC/CDT 79-005
Rev 88

Canada

Transports Canada Transport Canada

FORMULE DE DONNÉES POUR PUBLICATION

1. N° de la publication de Transports Canada	2. N° de l'étude	3. N° de catalogue du destinataire
TP 9990E	7049	

4. Titre et sous-titre

CORES: Air Traffic Conflict Resolution System
Final Report

5. Date de la publication
Octobre 1989

6. N° du document de l'organisme exécutant
—

7. Auteur(s)
I.G. Smith, G.R. Ram, M.R. Attisha, C.A. Crabb

8. N° de dossier — Transports Canada
D 1455-200-1

9. Nom et adresse de l'organisme exécutant
CompEngServ Ltd.
265 avenue Carling, suite 600
Ottawa (Ontario)
K1S 2E1

10. N° de dossier — ASC
11SD.T8200-7-7562

11. N° de contrat — ASC ou Transports Canada
T8200-7-7562/01-SD

12. Nom et adresse de l'organisme parrain
Centre de développement des transports (CDT)
Complexe Guy-Favreau
200 ouest, boul. René-Lévesque
Tour ouest, suite 601
Montréal (Québec) H2Z 1X4

13. Genre de publication et période visée
Rapport final,
juin 1988 – août 1989

14. Agent de projet
James D. Reid

15. Remarques additionnelles (Programmes de financement, titres de publications connexes, etc.)
CORES User's Guide, Testing Document, et Software Design, volumes 1 à 5 sont publiés sous couverture séparée.

16. Résumé

Description du prototype d'un système d'aide à la résolution des conflits en contrôle de la circulation aérienne, baptisé CORES (COnflict REsolution System). Il s'agit d'un système autonome permettant de simuler les mouvements des aéronefs dans un espace aérien, d'en identifier les incompatibilités potentielles, de suggérer des parades et d'observer le résultat des procédures adoptées. Il est fondé sur un environnement non contrôlé par radar comme dans l'espace aérien du Canada septentrional. CORES a montré la faisabilité des systèmes automatiques d'aide à la résolution de conflits de circulation aérienne dont il permet de tester les possibilités dans des conditions d'exploitation très variées. Le rapport décrit le système, donne le résultat de l'évaluation faite et propose des recherches en vue de son développement ultérieur.

17. Mots clés	18. Diffusion
Contrôle de la circulation aérienne, résolution de conflits, systèmes experts.	Le Centre de développement des transports dispose d'un nombre limité d'exemplaires.

19. Classification de sécurité (de cette publication)	20. Classification de sécurité (de cette page)	21. Déclassification (date)	22. Nombre de pages	23. Prix
Non classifiée	Non classifiée	—	49	—

CDT/TDC 79-005
Rev 88

Canada

SUMMARY

1. INTRODUCTION

In 1988, CompEngServ Ltd. of Ottawa was tasked by the Transportation Development Centre of Transport Canada to develop a prototype expert system advisor for air traffic conflict resolution. The objective of developing this prototype, known as CORES (COnflict REsolution System), was to assist the Air Traffic Control Research and Experimentation Centre in the examination of new ATC tools and training vehicles to be considered for the future workstation currently under study.

The design of CORES was based on a previous prototype conflict resolution system and proposed extensions to this prototype. The previous prototype was incomplete in that only certain types of conflicts were handled and many of the classical algorithms were missing (e.g., computing the details of specific maneuvers).

In addition to completing the missing portions of the previous prototype, the primary goal was to further develop and refine the proposed extensions in order to solve the problem of convergence. Convergence refers to the problem of an airspace containing some number of aircraft, where the result of maneuvering any one aircraft may have secondary effects (induced conflicts) which require more aircraft to be maneuvered. In this type of situation, it is not clear that the process of maneuvering aircraft can *converge* to a completely conflict-free airspace due, in theory, to the exponential complexity of the problem. However, in practical terms the goal of always achieving a completely conflict-free airspace is unnecessary. Since there is inherent uncertainty in any long term predictions of aircraft positions, it may be acceptable to induce an uncertain conflict (e.g., one which is several hours away) in order to resolve an immediate conflict.

2. SCOPE

In order to develop, test, and experiment with the conflict resolution algorithm, a stand-alone demonstration package was constructed which can graphically illustrate the use of the resolution algorithm. The original prototype was built using an expert system builder that was deemed too limited to achieve the goals of the project. Instead, portions of the original prototype were reimplemented within CORES which was built using the 386 version of the KEE expert system builder from Intellicorp. Discussion of the system is divided into two areas, the demonstration system and the conflict resolution algorithm itself.

It is important to keep in mind that CORES is only an advisory system; the controller is still an integral part of the conflict resolution process. Under the current air traffic control environment there are data which are not readily available to an automated system, such as information found only in direct pilot/controller voice communications and the exact

navigation capabilities of an aircraft. CORES will suggest solutions that appear to be feasible using the available information. It is up to the controller to make the final judgement on whether any given solution is appropriate.

3. CORES

The demonstration system provides the simulation capabilities necessary to achieve the goals of the project. It was not necessary to build a fully functional simulator, and as such several assumptions and simplifications have been made.

CORES is a stand-alone system which models a hypothetical airspace. This airspace was specially constructed to allow difficult scenarios involving crossing-track traffic to be developed. It contains both predefined INS tracks and routes based on NAVAIDs. The system allows the creation and observation of conflict scenarios in this airspace. It also allows the observation of the automated procedures that provide the resolution advice.

The design of the conflict resolution algorithm was guided by three underlying principles:

i) Minimal perturbation of the airspace: maneuver as few planes as possible and keep deviations from the preferred flight plans to a minimum.

ii) ATC constraints: consider such things as the airspace structure, aircraft performance characteristics, and the number of times a plane has been previously maneuvered when deciding on resolution procedures.

iii) Widening field of view: In a similar fashion to a controller, consider only the immediate factors involved with the predicted conflict first, followed by more global factors as necessary.

These three principles have been embodied in a multi-stage pipeline style process. The first three stages (initial resolution, relax constraints, and convergence) deal with conflict situations of increasing complexity. At each of these stages, ways of avoiding the predicted conflict can be suggested.

Initial Resolution Stage

The initial resolution stage classifies the predicted conflict as one of six basic types and selects strategies to test accordingly. The strategies tested during this stage are those that would typically resolve the majority of conflicts encountered. If a strategy can be executed such that no induced conflicts remain in the airspace then it is successful.

Relax Constraints Stage

Using those strategies that were tested by the initial stage but which failed, the relax constraints stage will attempt more extreme strategies. An example of a more extreme strategy would be routing a plane to a wrong-way altitude. Typically, the minimal perturbation constraints are relaxed during this stage.

Convergence Stage

Finally, the convergence stage examines all those strategies which previously failed because they induced one or more further conflicts. In the convergence stage it is assumed that no strategies exist which produce a completely conflict-free airspace. The goal of this stage is to suggest partial solutions which result in a better overall airspace in terms of the criticality and difficulty of the induced conflicts. If the convergence stage fails to find any solutions which result in an improved airspace, the system will call the controller for help. Under these circumstances the system will provide the controller with all of the information known about why strategies have failed.

Unlike other automated conflict prediction mechanisms, which typically test aircraft separation using constant or variable sized circles drawn around each aircraft, the conflict prediction mechanism in CORES uses a myriad of ATC procedural separation rules. This requires the prediction mechanism to fully analyze the conflict situation in order to recognize which rules to apply (e.g., crossing track versus same direction rules).

4. RESULTS

CORES has demonstrated the feasibility of an operational conflict resolution advisory system. The resolution advice presented by the system is reasonable under the constraints of the system. Criticism included the length of time required to produce solutions, and the ranking that was performed, but not the type of advice produced.

In testing the limitations of the resolution algorithm, the convergence stage produced quite reasonable advice, while the call for help stage was rarely exercised. In fact, it was difficult to find a realistic scenario to test this portion of the algorithm. (The scenario used for testing is somewhat unrealistic, but possible.)

CORES was developed using the 386 version of the KEE expert system builder from Intellicorp. This version of 386-KEE (the August 29, 1988 release of version 3.1) fell below the expectations and requirements of the project. The KEE-related problems and limitations which severely affect the performance and impact of the system include, memory management deficiencies, graphics performance and resolution problems, and lack of multitasking facilities.

Results of this work have also included a number of possibilities for extensions to CORES itself, or to these types of systems. These are outlined below, and described in more detail in Section 5 of this report.

5. RECOMMENDATIONS

It has been suggested that investigation into conflict resolution systems be continued; this could include functions such as, adapting CORES for use in general flight planning applications, evaluating controller workload based on route structures or procedural changes, and the design of a resolution system for a tactical radar environment. Further work on CORES itself could include investigations into the man-machine interface issues involved with presenting resolution advice, expanding the number of strategies to include more of the possible solutions used by controllers, or allowing the dynamic modification of flight plans while the demonstration is running.

SOMMAIRE

1. INTRODUCTION

En 1988, le Centre de développement des transports de Transports Canada a mandaté la firme CompEngServ Ltd. d'Ottawa afin d'étudier le prototype d'un système expert pour aider les contrôleurs à résoudre les conflits de circulation aérienne. Ce projet de recherche baptisé CORES (COnflict REsolution System) devait permettre au Centre de recherche et d'expérimentation des services de la circulation aérienne de tester les possibilités des nouveaux outils de formation ATC en matière de contrôle de la circulation aérienne en vue de leur intégration dans le poste de travail actuellement à l'étude.

Le prototype CORES est fondé sur un modèle antérieur et sur les extensions qui avaient été proposées pour ce dernier, étant donné qu'il était incomplet dans ce sens qu'il intéressait certains types de conflits seulement et qu'il ne comportait pas plusieurs des algorithmes classiques, comme ceux qui décrivent de façon détaillée la manière d'effectuer certains types de transferts, par exemple.

En plus de combler les lacunes du prototype antérieur, la recherche lançée visait à développer et affiner davantage les extensions proposées en vue de résoudre les cas de convergence. Par convergence dans le domaine du contrôle de la circulation aérienne, on entend les cas où le transfert d'un aéronef se trouvant dans un espace aérien donné avec d'autres peut entraîner le transfert d'autres aéronefs, provoquant ce qu'il est convenu d'appeler des conflits induits. Dans ce type de situation, il n'est pas évident que la décision d'opérer un transfert aboutisse nécessairement à un espace aérien exempt de conflits, la raison étant, en théorie, que ce type de problème s'amplifie exponentiellement; mais il faut bien préciser que, dans la pratique, il n'est pas toujours indispensable d'avoir un espace aérien completement exempt de tout conflit. Vu les incertitudes inhérentes à toutes les prévisions à long terme des positions occupées par les aéronefs dans un espace aérien donné, un conflit induit de type aléatoire, c'est-à-dire qui pourrait se produire beaucoup plus tard, serait acceptable s'il permettait de résoudre un conflit plus immédiat.

2. PORTEE

Dans le but de réaliser, d'essayer et d'expérimenter l'algorithme de résolution des conflits de circulation aérienne, un ensemble de démonstration autonome a été réalisé, utilisant certains éléments du prototype antérieur, jugé insuffisant, et intégrant la version 386 d'un système expert élaboré à l'aide du générateur de système expert KEE d'Intellicorp. La description de ce système distingue le système proprement dit de l'algorithme qui l'anime.

Il ne faut pas oublier que CORES a été conçu comme une aide à la décision et non en vue de supplanter le contrôleur qui reste toujours dans la boucle ATC. Dans le contexte actuel, certaines sources d'information ne se prêtent pas encore à l'automatisation, exemple les communications radiotéléphoniques pilote-contrôleur et les caractéristiques particulières des aéronefs. C'est pourquoi le but fixé au CORES est de proposer des solutions à partir des informations qu'il a en mémoire et il appartiendra au contrôleur, d'après son expérience, de les accepter ou de ne pas en tenir compte.

3. CORES

La démonstration doit servir à illustrer les aptitudes de CORES à la simulation et, pour sa réalisation, certaines hypothèses ont été retenues et certaines simplifications faites, étant donné que l'objectif n'était pas de construire un simulateur complètement fonctionnel.

CORES est un système autonome permettant de modéliser un espace aérien hypothétique, mais conçu de manière à pouvoir reconstituer certaines situations délicates telles que des trajectoires qui se croisent, de les étudier et d'observer le résultat des procédures suggérées à titre de solutions possibles. Dans le système sont modélisées les voies et les trajectoires prédéfinies de navigation inertielle et d'autres axées sur les aides à la navigation.

L'algorithme a été élaboré selon les trois principes suivants:

i) Minimaliser les pénalisations: faire le moins de transferts possible et réduire au maximum l'écart par rapport aux plans de vol établis;

ii) Tenir compte des contraintes ATC: organisation de l'espace aérien, caractéristiques des aéronefs et nombre de transferts dont un aéronef a été l'objet;

iii) Classifier les conflits prévisibles: à l'instar du contrôleur, prendre en considération d'abord les facteurs immédiatement en jeu touchant le conflit qui se dessine et ensuite, selon les circonstances, les autres facteurs plus lointains.

Ces trois principes ont été intégrés dans une architecture en série comportant plusieurs étapes, dont les trois premières - classification des conflits, relaxation des contraintes et convergence - traitent de situations de complexité grandissante auxquelles des solutions peuvent être proposées.

Classification des conflits

Dans cette première étape, les conflits prévisibles sont classés selon l'une des six classes connues et là ou les stratégies adoptées pour les résoudre sont testées. Les stratégies testées à cette étape sont celles qui sont susceptibles de résoudre la plupart des conflits potentiels. Sera considérée comme ayant réussi celle qui aura permis de résoudre tous les conflits sans qu'il subsiste des conflits induits.

Relaxation des contraintes

L'étape suivante consiste à mettre à l'épreuve les stratégies précédentes qui auraient échoué, mais en les poussant à l'extrême tel que par exemple d'affecter à un aéronef une altitude contre-indiquée par la direction dans laquelle il vole. Dans ces cas, la contrainte de pénalisation minimale est généralement relaxée.

Convergence

Dans cette dernière étape sont examinées les stratégies ayant raté parce qu'elles ont provoqué un conflit induit ou plus. Il est d'ores et déjà admis qu'il n'existe pas de stratégies permettant d'éliminer complètement les conflits. Le but de cette étape est donc de trouver des solutions ne serait-ce que partielles dont la mise en oeuvre aboutirait à un espace aérien amélioré, c'est-à-dire dénué de conflits induits du genre immédiat et compliqué. A défaut de telles solutions, le système s'avouera vaincu et alertera le contrôleur, qu'il mettra au courant des stratégies essayées et des raisons de leur échec.

Contrairement aux autres systèmes automatiques de gestion des conflits, qui typiquement vérifient les distances de séparation en inscrivant les aéronefs au centre d'un cercle de diamètre uniforme ou variable, le système CORES utilise une multitude de règles de séparation en vigueur, ce qui suppose qu'il est en mesure d'analyser à fond le conflit et d'identifier la ou les règles applicables, par exemple, aux trajectoires qui se croisent ou qui sont parallèles dans la même direction.

4. RESULTATS

Le système CORES a montré la faisabilité des systèmes d'aide à la résolution de conflits de circulation aérienne, les solutions qu'il propose étant tenues pour raisonnables dans le contexte contraignant choisi. Les faiblesses reprochées comprenaient le temps de réponse élevé et le classement des conflits mais pas la solution proprement dite.

Poussé au bout de ses capacités durant l'étape de convergence, l'algorithme a donné des résultats jugés très adéquats et n'a que rarement déclaré forfait. En fait, il a été difficile d'écrire un scénario tel qu'il puisse prendre cet algorithme en défaut, le scénario en question étant considéré comme possible mais quelque peu aberrant.

CORES a été édifié autour de la version 386 du générateur de système expert KEE d'Intellicorp, divulgué le 29 août 1988 (version 3.1) mais qui n'a pas donné les résultats escomptés. Les faiblesses et les lacunes reliées au logiciel KEE concernent la gestion de la mémoire, les performances et la résolution infographiques et l'absence de fonctions multitâches.

La recherche a permis de définir des possibilités d'extension du système CORES proprement dit ou de systèmes analogues. Les extensions proposées sont esquissées ci-dessous et décrites en détail dans le rapport.

5. RECOMMANDATIONS

Le rapport propose de poursuivre la recherche sur les systèmes de résolution des conflits de circulation aérienne, notamment par l'adjonction des nouvelles fonctions suivantes: adaptation du CORES à la planification générale des vols, à l'évaluation de la charge de travail du contrôleur suscitée par des changements proposés à la structure des voies aériennes ou aux procédures prescrites et à l'étude d'un système adapté à un contrôle radar de surveillance tactique. Des recherches plus poussées intéressant le système proprement dit pourraient s'attaquer à l'interface contrôleur-système et plus particulièrement à la présentation des solutions, à l'augmentation du nombre de stratégies envisageables et, donc, des solutions possibles ou à la possibilité de modifier les plans de vol pendant le déroulement du scénario.

TABLE OF CONTENTS

	Page
1. INTRODUCTION	1
1.1 Air Traffic Control in Canada	1
1.2 Project Objectives and Scope	1
1.3 Historical Overview	2
1.4 Special Terminology	3
1.5 Organization of Report	4
2. CORES	5
2.1 Demonstration System	5
2.1.1 Modes of Operation	5
2.1.2 Demonstration Cycle	7
2.1.3 User Interface	12
2.1.4 Additional Features	14
2.2 Algorithm	15
2.2.1 Overview	15
2.2.2 Initial Resolution Stage	17
2.2.3 Relax Constraints Stage	19
2.2.4 Convergence Stage	21
2.3 Assumptions	23
2.4 Miscellaneous Information	24
3. EVALUATION	26
3.1 KEE Performance Limitations	26
3.2 Algorithm Tests	27
3.2.1 Advice Produced by the Algorithm	28
3.2.2 Algorithm Performance	28
3.2.3 Conflict Prediction Performance	30
3.2.4 Algorithm Design Issues	30
3.2.5 Unresolved Man-Machine Interface Issues	31
3.3 Demonstration System	32
4. CONCLUSIONS	33
5. RECOMMENDATIONS	34
6. REFERENCES	36
7. BIBLIOGRAPHY	37

The following documents are published under separate cover:

CORES: User's Guide,

CORES: Testing Document,

CORES: Software Design Documents (Volumes 1 through 5).

FIGURES

	Page
Figure 1 CORES Airspace	6
Figure 2 CORES Update Screen	8
Figure 3 The CORES Demonstration Cycle	10
Figure 4 CORES Solutions Menu	11
Figure 5 CORES Main Screen	13
Figure 6 Main Stages of the Conflict Resolution Algorithm	16
Figure 7 Initial Resolution Stage	18
Figure 8 Relax Constraints Stage	20
Figure 9 Convergence Stage	22

GLOSSARY

This section lists, with definitions, all special terms and acronyms used in this report.

AGARD	Advisory Group for Aerospace Research and Development.
AI	Artificial Intelligence.
ATC	Air Traffic Control.
ATSAMM	Air Traffic Services Administration and Management Manual, Transport Canada
CORES	Conflict Resolution System; developed by CompEngServ Ltd. in 1988-1989.
MANOPS	Manual of Operations, Air Traffic Services, Transport Canada
NATO	North Atlantic Treaty Organization.
TDC	Transportation Development Centre.

1. INTRODUCTION

This document describes a proof-of-concept advisory system for air traffic conflict resolution, known as CORES (COnflict REsolution System). The purpose of this document is to provide an overview of CORES, an evaluation of the system, and recommendations for future work. This document is written as a stand-alone report on the system. For further details of the use, design, or testing of CORES refer to [1], [2], and [3].

1.1 Air Traffic Control in Canada

Demands on air traffic control (ATC) facilities throughout the world are ever increasing. As a result, controllers are in need of more sophisticated tracking and planning aids to provide the expeditious flow that the airlines demand, while still maintaining a high degree of safety. Artificial intelligence (AI) techniques are being investigated to provide some of these aids.

Due in part to its vast size, most of Canada's northern airspace is without civilian radar coverage. Currently, non-radar airspace is controlled using only aircraft flight plans and periodic position reports from aircraft. This requires a controller to keep a precise mental picture of the airspace at all times with the additional aid of a manual or electronic plotting board. Because of such factors as weather and distances between waypoints, there is ample room for error in estimating the positions of aircraft. Understandably, aircraft separation criteria are particularly strict in such an environment. Compounding all of these factors is Canada's geographic location as a convergence point for many overseas flights.

An automated conflict prediction and conflict resolution advisory system would be of obvious benefit to controllers. However, before the design and production of such a system can occur, a suitable conflict resolution algorithm must be developed and proven useful. Careful integration with other ATC systems and extending the capabilities of peripheral functions is of no use if the conflict resolution algorithm itself produces poor or unreliable advice.

1.2 Project Objectives and Scope

The objective of the CORES project was to develop a prototype conflict resolution advisory system to assist the Air Traffic Control Research and Experimentation Centre in the examination of new ATC tools and training vehicles to be considered for the future workstation currently under study.

The design of CORES was based on a previous prototype conflict resolution system (described in [1, 2, 3, and 4]) and proposed extensions to this prototype as reported in [5]. The previous prototype was incomplete in that only certain types of conflicts were handled and many of the classical algorithm routines were missing (e.g., computing the details of specific maneuvers).

In addition to completing the missing portions of the previous prototype, the primary goal was to further develop and refine the extensions described in [5] in order to solve the problem of convergence. Convergence refers to the problem of an airspace containing some number of aircraft, where the result of maneuvering any one aircraft may have secondary effects (induced conflicts) which require more aircraft to be maneuvered. In this type of situation, it is not clear that the process of maneuvering aircraft can *converge* to a completely conflict-free airspace due, in theory, to the exponential complexity of the problem. However, in practical terms the goal of always achieving a completely conflict-free airspace is unnecessary. Since there is inherent uncertainty in any long term predictions of aircraft positions, it may be acceptable to induce an uncertain conflict (e.g., one which is several hours away) in order to resolve an immediate conflict.

In order to develop, test, and experiment with the conflict resolution algorithm, a stand-alone demonstration package was constructed which can graphically illustrate the use of the resolution algorithm (see Section 2.1 for more details). The original prototype was built using an expert system builder that was deemed too limited to achieve the above goals. Instead, portions of the original prototype were reimplemented within CORES which was built using the 386 version of the KEE expert system builder from Intellicorp.

1.3 Historical Overview

As the result of an unsolicited proposal to the Transportation Development Centre (TDC) in 1985, CompEngServ Ltd. designed and built a small prototype expert system (as reported in [1]). The purpose of the prototype was to show that expert systems technology can be applied to the problem of conflict avoidance (under the limiting assumption that a conflict may be avoided without causing another conflict). At the conclusion of this work, it was proposed that many of the concepts developed could be used in the solution of the conflict avoidance problem in arbitrarily complex airspaces.

In 1987, Dr. B.A. Bowen of CompEngServ Ltd. was invited by the Advisory Group for Aerospace Research and Development (AGARD) of NATO, to present a paper which discussed extensions to the original work. These extensions were proposed in order to deal with situations where a resolution procedure induces further conflicts which in turn have to be resolved (i.e., the convergence problem). The extensions proposed were three additional steps to be performed if no solution could be found which provided a conflict-free airspace. The first of these steps is to relax constraints in order to attempt to increase the possible number of strategies to test (exactly which constraints to be relaxed is not specified). The second step is to partition the strategies into critical and non-critical based on whether the induced conflicts can be resolved within the response time of the system. Finally the third step is to call the controller for help if all possible solutions increase the total number of conflicts in the airspace.

In 1988, CompEngServ Ltd. received a contract, described in this report, to further refine and implement the extensions described in the AGARD/NATO paper [5]. Although the same general steps have been used in the implementation (initial resolution, relax constraints, partition, and call for help), the details of what the steps perform and how they achieve their results are substantially different from those described in the AGARD/NATO paper. Refer to Section 2.2 for a description of the algorithm which was implemented.

1.4 Special Terminology

As discussed above, the design of CORES is partially based on the algorithm and extensions described in [5]. However, there are several important differences between the terminology used in the AGARD/NATO paper [5], the terminology used by ATC personnel, and the terminology used here. We believe the terminology used here is consistent with ATC terminology. The definitions of the relevant terms are as follows:

Sparse Airspace:

ATC: usually implies non-radar airspace.

AGARD/NATO paper: is one in which a resolution is effected without inducing further conflicts. Also refers to the first stage of the resolution algorithm presented in the paper.

CORES: term is not used. The first stage of the resolution algorithm is referred to as the Initial Resolution Stage. In this stage induced conflicts are considered but not resolved.

Other: in some research papers the term "sparse airspace assumption" is used. This usually implies that no tests or action for induced conflicts is undertaken.

Medium Density Airspace:

ATC: usually implies enroute airspace with radar coverage (not a terminal environment).

AGARD/NATO paper: term not used.

CORES: term not used.

Dense Airspace:

ATC: usually implies terminal environment.

AGARD/NATO paper: is one in which a resolution tactic could induce subsequent conflicts. Also used to describe all the other stages of the algorithm not covered under the definition of sparse.

CORES: term not used. The final stages of the algorithm are referred to separately as "relax constraints", "convergence", and "call for help".

1.5 Organization of Report

Section 2 provides an overview of CORES functionality and a summary of the conflict resolution algorithm. Section 3 is an evaluation of CORES. The evaluation is divided into a discussion of the resolution algorithm performance and a discussion of the demonstration tool functionality. Section 4 summarizes the evaluation. Section 5, discusses further refinements and possible extensions of this work to other applications.

2. CORES

This section presents an overview of CORES functionality, a closer look at the conflict resolution algorithm used within CORES, and a list of the major assumptions the system makes use of. A complete description of CORES functionality (i.e., how to use the demonstration system) can be found in the CORES: User's Guide [1]. A more detailed description of the conflict resolution algorithm can be found in Appendix F of the User's Guide.

It is important to keep in mind that CORES is only an advisory system; the controller is still an integral part of the conflict resolution process. Under the current air traffic control environment, there are data which are not readily available to an automated system, such as information found only in direct pilot/controller voice communications and the exact navigation capabilities of an aircraft. CORES will suggest solutions that appear to be feasible using the available information. It is up to the controller to make the final judgement on whether any given solution is appropriate.

2.1 Demonstration System

The demonstration system was constructed in order to assess the effectiveness of the conflict resolution algorithm in dealing with conflicts of varying degrees of complexity. The demonstration system was not designed to be a complete prototype of an operational system.

The demonstration system provides the simulation capabilities necessary to achieve the above goals. It was not necessary to build a fully functional simulator; the assumptions and simplifications that have been made are described in Section 2.3.

CORES is a stand-alone system which models a hypothetical airspace (see Figure 1). This airspace was specially constructed to allow difficult scenarios involving crossing-track traffic to be developed. It contains both predefined INS tracks and routes based on NAVAIDs. The system allows the creation and observation of conflict scenarios in this airspace. It also allows the observation of the automated procedures that provide the resolution advice.

2.1.1 Modes of Operation

The system consists of three conceptual modes of operation: Update, Running and Solving. Update mode refers to the *a priori* creation or modification of scenarios used to demonstrate the resolution algorithm. This data consists of the expected aircraft flight paths through the defined airspace, and those events that will occur unexpectedly to aircraft during their flights causing them to come off-schedule.

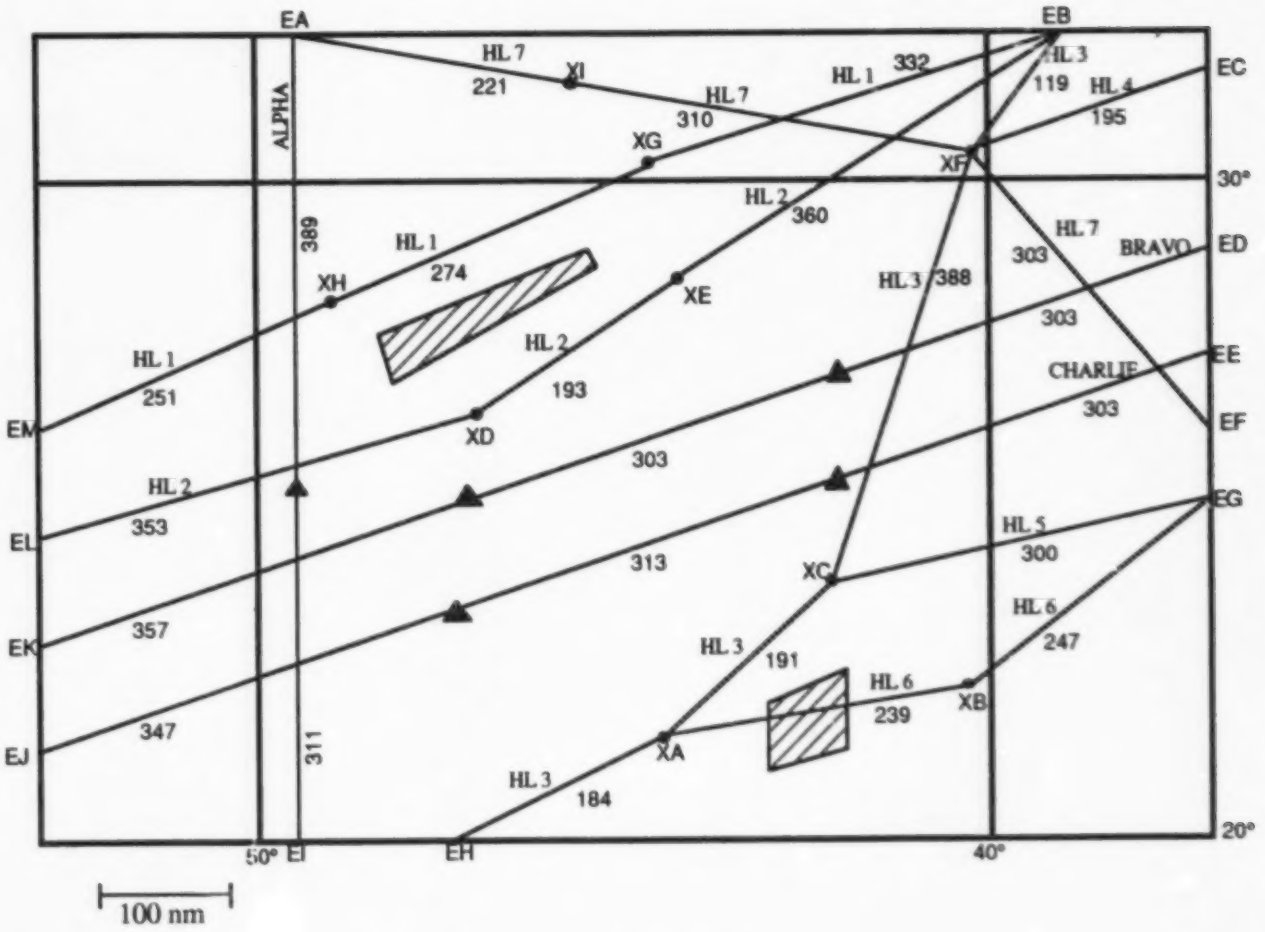


Figure 1: CORES Airspace

In Running mode, CORES displays the progress of aircraft through the hypothetical airspace according to predefined data (created or modified in Update mode). Whenever an event causes an aircraft to come off-schedule or whenever a new aircraft enters the airspace, CORES will automatically test for potential future losses of separation (conflicts). The entire flight paths of aircraft are tested for potential conflicts from their current position until the point where they leave the airspace.

The third mode, Solving, is entered whenever a conflict is predicted. This mode is devoted to producing conflict resolution advice and the examination of the advice and the process used to produce the advice. While in this mode, the progress and display of aircraft (Running mode) is stopped.

2.1.2 Demonstration Cycle

In Update mode, two mechanisms are provided to enable flight description: flight plans and events. Each plane has a flight plan and a list of events that occur during the course of its passage through the airspace. Like ATC flight plans, CORES flight plans detail where a plane will go and when it will be there. As well as specifying the route that the plane will take and its ground speed, the user can also enter such data as flight levels, aircraft types, airspeed, and whether or not the flight is a pop up. Events represent changes to aircraft flight plans that occur dynamically during flight and can be thought of as pilot position reports. As such, an event can alter the current estimated position of an aircraft (i.e., an aircraft is so many minutes ahead or behind schedule), or the expected time of arrival (ETA) at the next flight plan position (i.e., the ground speed is so many knots above or below the planned speed). Additionally, events can be used to simulate the effects of changing winds by creating events for all planes on a given track or airway.

Figure 2 below shows an example entry to the CORES Update Screen. Here, flight KLM123's flight plan is only partially entered: only its entry point is recorded. The plane is a Boeing 727 flying at 450 knots, that will pop up into the airspace at fix EL at 1114h. At 1227h, KLM123 will experience a 20 kts decrease in ground speed. Positions and events can be added, changed and deleted using the appropriate menu options at the top of the screen. When the user has completed entry of the plane's flight plan, selecting SAVE will store the newly created flight plan into the systems Flight Plan and Events Database. Chosing EXIT on the CORES Update Screen will return the user to the CORES Main Screen.

Several features are included in the Update function to facilitate the creation of conflict situations. For example, arrival times at fixes are continually updated during the flight plan creation process. This allows the user to plan conflicts more efficiently by having important data crucial to the conflict condition readily available. In addition, error checking processes ensure that users do not inadvertently submit unrealistic flight plans.

Once a suitable number of scenarios have been created (this may only be one), the user proceeds to the second mode: Running. Before doing this, the SET CLOCK option on the CORES Main Screen can be selected in order to set the demonstration time to the time the first plane in the scenario enters. As only planes that enter the airspace after the initial

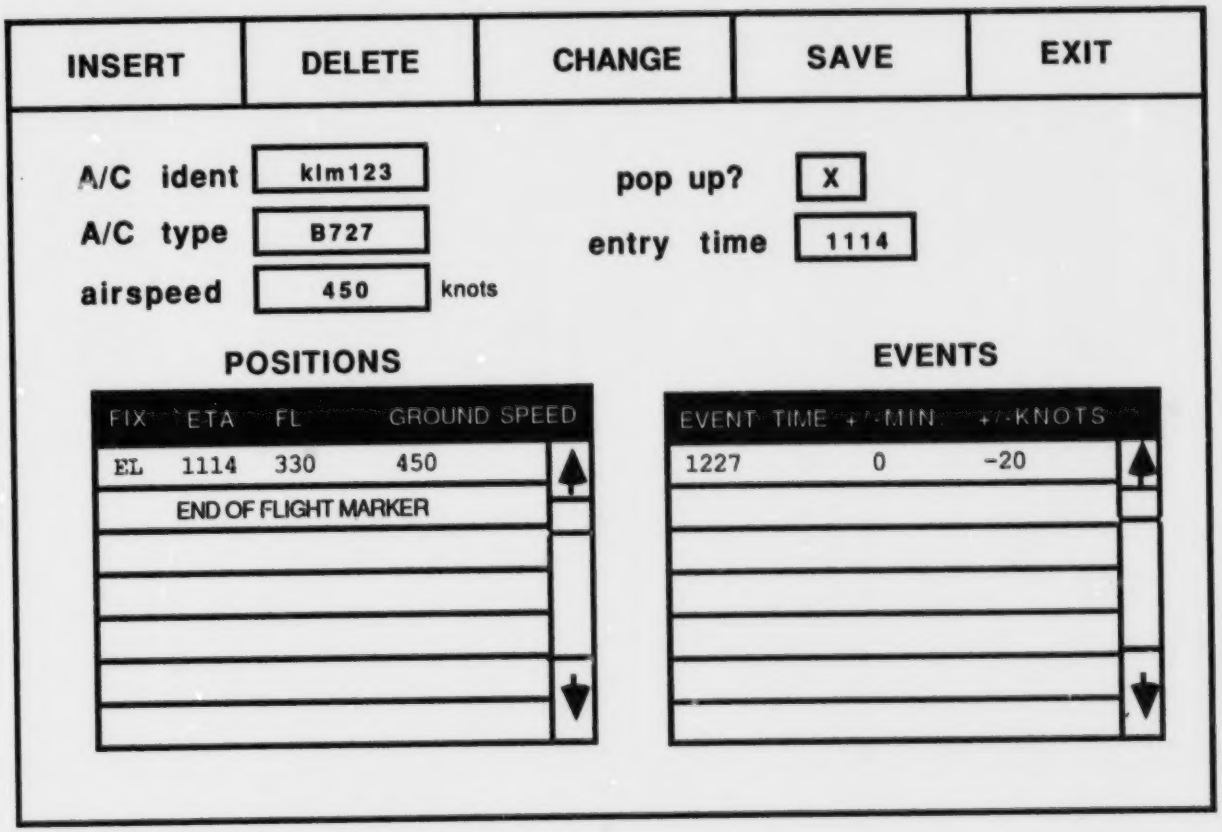


Figure 2: CORES Update Screen

demonstration time are considered during the Running mode, it is often necessary to select the demonstration time carefully in order to avoid unwanted conflicts. To enter Running mode, RUN DEMONSTRATION is selected from the Main System Menu.

The planes are graphically depicted and moved according to their flight plans and the events that are enacted upon them. This enables the user to monitor the relative positions of planes as they move through the demonstration airspace. This in turn facilitates the construction of conflict scenarios. For example, should the user be dissatisfied with the current scenario, the demonstration can be exited, and the user can re-enter Update mode to alter the required flight plan(s). Following this, the demonstration can be restarted to monitor the newly constructed scenario.

The relationship between the Running and Solving portions of the Demonstration System can be conveniently depicted with the Demonstration Cycle shown in Figure 3. On each iteration of the Demonstration Cycle, the demonstration clock that drives the demonstration is incremented. Planes are then moved based on their flight plans to their expected positions at the updated time.

Once all planes are moved, any events which have arisen, such as a plane entering the airspace or reporting a change in speed, will be executed. If no such event exists, the demonstration clock is incremented and the demonstration cycle continued. Should an event exist, the plane incurring the event is tested through the entire airspace, and against all other planes and restricted airspaces, to determine if any potential conflicts have arisen. If not, the demonstration clock is incremented and the cycle is continued. However, should a potential conflict be detected, the system begins to solve it; this corresponds to the Solving mode. During the resolution process, messages identifying the current state of the solving procedure, are displayed at the top of the screen. Furthermore, while in Solving mode, the Running mode of the demonstration system halts, as the algorithm attempts to solve the predicted conflict.

When the system has determined a set of resolution procedures for a predicted conflict, the solutions are ranked and presented to the user via the CORES Solutions Menu. For example, in Figure 4, an overtaking conflict between ACA771 and KLM123 produced six different solutions which are ranked, and include such maneuvers as descending ACA771 to FL290 (the most preferred) and rerouting KLM123 (the least preferred).

The user can choose any one of these solutions and analyze what would happen if it were executed by selecting 'what if?'. Similarly, by selecting 'details' the user can see the plane's original flight plan, and the modified version which incorporates the corresponding maneuver. Finally, selecting 'why?' will produce a logic trace of how the solutions were determined. These features were designed so the user can validate the correctness of the solutions.

Once a particular resolution procedure is selected, and the user chooses 'execute', system control is returned to the process Increment Clock, and the demonstration cycle is continued.

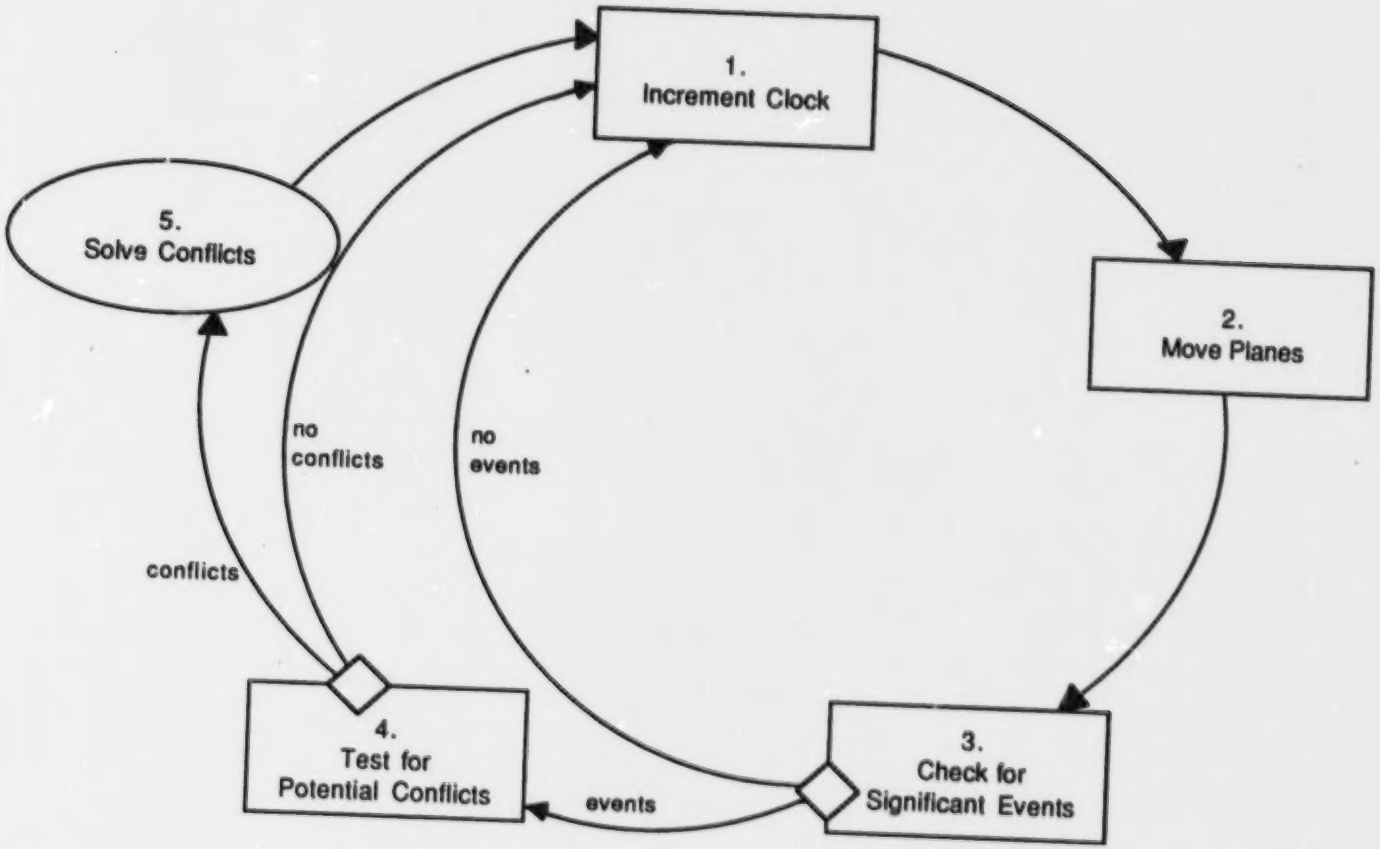


Figure 3: The CORES Demonstration Cycle

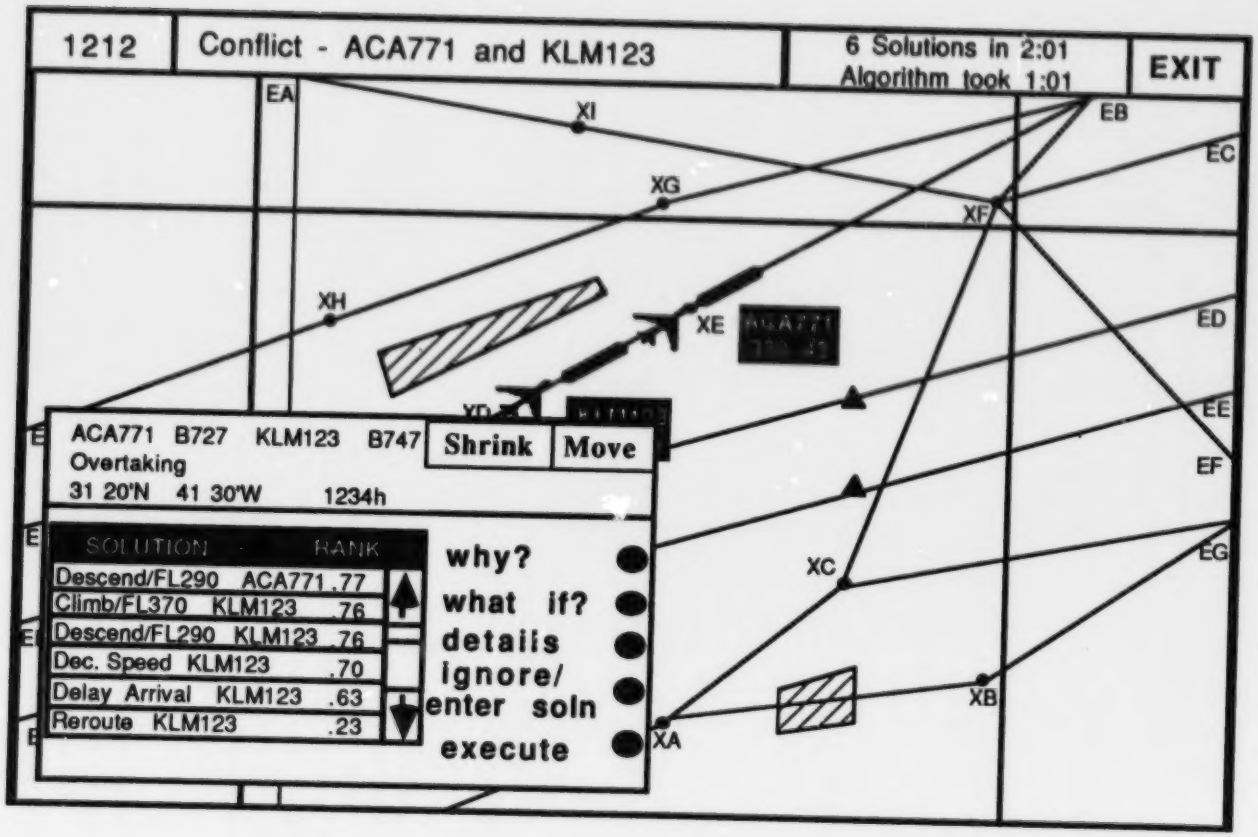


Figure 4: CORES Solutions Menu

The cycle continues in this manner until the user selects the EXIT option, there are no more planes to enter the airspace, or the demonstration clock reaches 2359h.

At the end of a demonstration run, a history log of the session can be obtained by selecting the VIEW LOG option on the Main System Menu. This produces a scrollable window that contains an abbreviated transcript of the last demonstration session, which includes information regarding flight entry, events, conflict detection, and solutions.

2.1.3 User Interface

The CORES user interface is menu driven, and designed to allow the user to test the conflict resolution algorithm. The system begins with the CORES main screen (see Figure 5), which contains an introduction to CORES, displayed mainly to ensure the system is being used as intended; e.g., the user does not run the demonstration believing CORES to be an operational system. Also included is the main menu, at which point the user can choose to run the demonstration as it has been set up, update the flight plans and events, set the clock to begin the demonstration at a different time, or view a history log of the last demonstration run.

When running the demonstration, the CORES airspace screen is displayed. This airspace was specially constructed to allow difficult scenarios involving crossing-track traffic to be developed. It contains both predefined INS tracks, and routes based on NAVAIDs. It also contains restricted airspaces which do not necessarily correspond to ATC restricted areas; they could simply identify severe weather conditions. The airspace is 900nm by 700nm, and extends from flight level 290 to flight level 410. Entering planes are displayed heading in their direction of flight, and associated with each is a tag displaying the aircraft identifier, the flight level, and the ground speed. An optional parameter allows the user to choose whether or not the leaders for the planes are displayed; however, turning the leaders on greatly slows the system down due to the graphics performance of KEE.

The demonstration screen includes the airspace screen, a system time display, the status of the demonstration cycle, and options to pause or exit the system. When a conflict is detected, the tags of the planes involved are highlighted, the conflict is identified in the status bar, the system clock is stopped, and the solutions screen is displayed with a gray background. While solving the conflict, the current processing being done by the algorithm is shown. When the resolution process is complete, the solutions screen background turns white, the solutions and ranks are presented in the solutions menu, and the time taken to solve the conflict is displayed (see Figure 4).

The solutions screen allows the user to check the logic behind the solutions found, or to see a graphical display of a chosen solution. The details of the solution, including the new flight plan, can also be shown. These features are to prove the validity of the solutions identified by the resolution algorithm. The other options available are ignore/enter solution to identify where an operational system would allow the user full control, and execute the chosen solution, which will also cause the demonstration to continue. The user can also shrink or move the solutions screen if it happens to be blocking the view of the conflict situation.

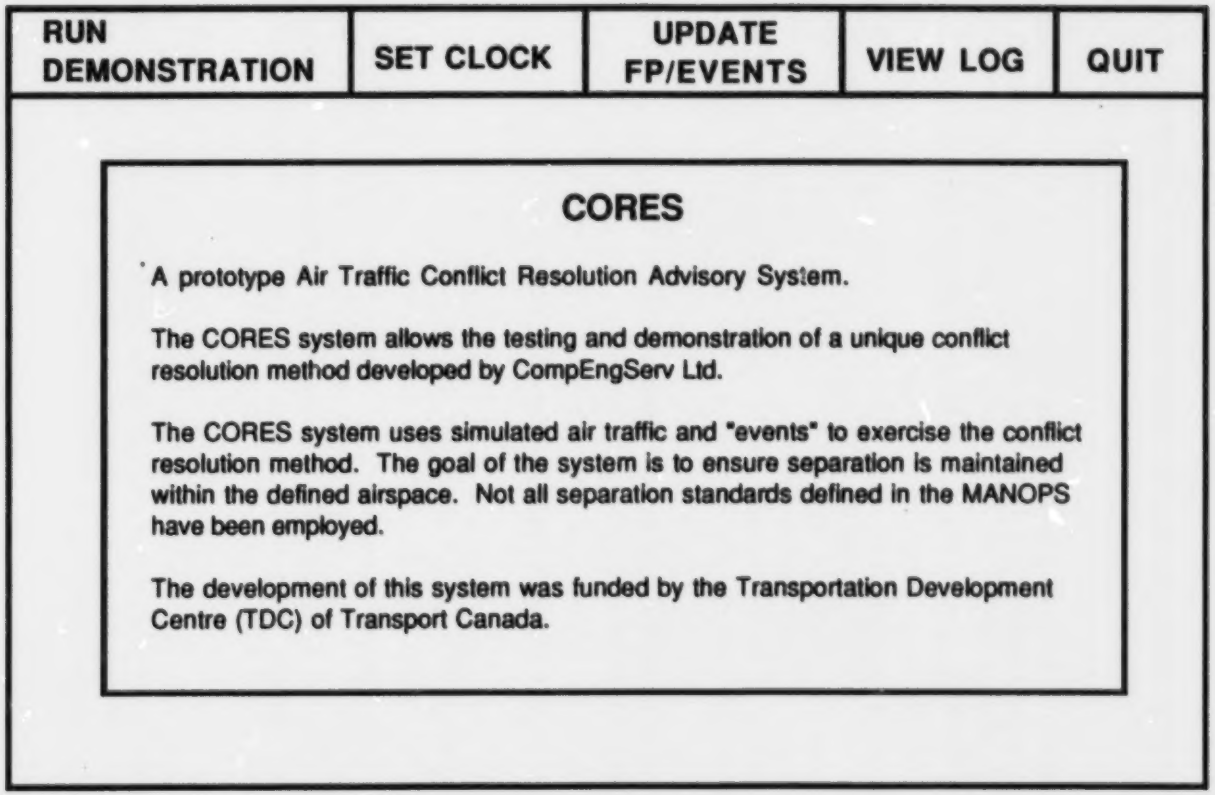

RUN DEMONSTRATION
SET CLOCK
UPDATE FP/EVENTS
VIEW LOG
QUIT
CORES
A prototype Air Traffic Conflict Resolution Advisory System.
The CORES system allows the testing and demonstration of a unique conflict resolution method developed by CompEngServ Ltd.
The CORES system uses simulated air traffic and "events" to exercise the conflict resolution method. The goal of the system is to ensure separation is maintained within the defined airspace. Not all separation standards defined in the MANOPS have been employed.
The development of this system was funded by the Transportation Development Centre (TDC) of Transport Canada.

Figure 5: CORES Main Screen

The system attempts to provide as much information as possible, for the user to effectively evaluate the performance of the conflict resolution algorithm.

2.1.4 Additional Features

The demonstration system provides the following features in addition to those described in the previous section:

- By default, the demonstration clock increments by one minute on each demonstration cycle. However, the clock increment can be adjusted by using the clock time-step feature provided with CORES. Using this feature, scenarios can be viewed at an accelerated rate.

- Planes can be displayed with leader lines, which indicate the longitudinal separation that must be maintained. The length of a plane's leader line is a function of the length of its flying time on the current leg and whether or not there are other planes on the leg. Leaders are adjusted automatically in CORES and are displayed with their correct length. For a discussion of how leaders are calculated, see Appendix E of the CORES User's Guide [1].

Several system parameters may be set before CORES is started using a special start-up file. The parameters are as follows:

- Turn the display of aircraft leaders on or off (default is off).

- Set the default system clock interval (normally 1 minute).

- Set the minimum number of solutions. The conflict resolution algorithm can be forced to continue finding additional solutions (see Section 2.2). Normally the algorithm will stop if solutions have been found in the initial solving stage (i.e., will not proceed to relax constraints). Note that the algorithm cannot be forced to perform convergence unless the conflict situation actually warrants its use.

- Set the maximum number of solutions. The conflict resolution algorithm can be forced to stop finding solutions in the middle of the initial resolution or relax constraints stages. Normally all possible solutions from the stage will be found. Note that solutions are pre-ranked before computing so that the best solutions should be found first (see Section 2.2).

- Force a garbage collection before solving. KEE uses "stop and copy" garbage collection. When the system is garbage collecting, CORES stops running. A garbage collection may take one or two minutes to complete. Because of this, the times for

solving may vary by large amounts depending on when garbage collections occur. To minimize this variance, the system may be forced to garbage collect early (immediately after a conflict is detected but before solving starts).

2.2 Algorithm

This section describes the conflict resolution algorithm used in CORES.

2.2.1 Overview

The design of the conflict resolution algorithm was guided by three underlying principles:

i) Minimal perturbation of the airspace: maneuver as few planes as possible and keep deviations from the preferred flight plans to a minimum.

ii) ATC constraints: consider such things as the airspace structure, aircraft performance characteristics, and the number of times a plane has been previously maneuvered when deciding on resolution procedures.

iii) Widening field of view: In a similar fashion to a controller, consider only the immediate factors involved with the predicted conflict first, followed by more global factors as necessary.

These three principles have been embodied in a multi-stage pipeline style process as shown in Figure 6. The first three stages (initial resolution, relax constraints, and convergence) deal with conflict situations of increasing complexity. Each of these stages can suggest ways of avoiding the predicted conflict.

Initial Resolution Stage

The initial resolution stage classifies the predicted conflict as one of six basic types and selects strategies to test accordingly. The strategies tested during this stage are those that would typically resolve the majority of conflicts encountered. If a strategy can be executed such that no induced conflicts remain in the airspace then it is successful.

Relax Constraints Stage

Using those strategies that were tested by the initial stage but which failed, the relax constraints stage will attempt more extreme strategies. An example of a more extreme

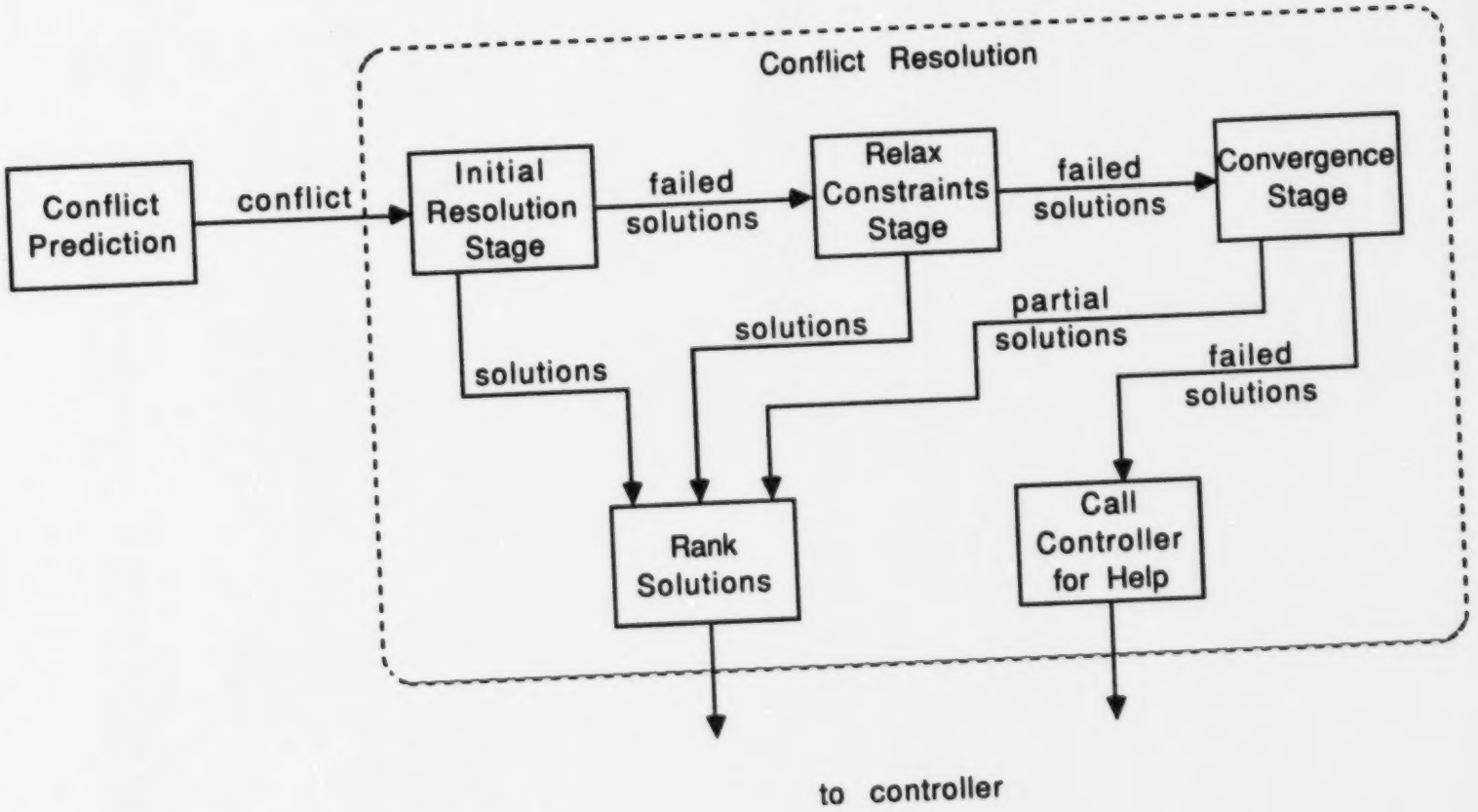


Figure 6: Main Stages of the Conflict Resolution Algorithm

strategy would be routing a plane to a wrong-way altitude. Typically, the minimal perturbation constraints are relaxed during this stage.

Convergence Stage

Finally, the convergence stage examines all those strategies which previously failed because they induced one or more further conflicts. In the convergence stage it is assumed that no strategies exist which produce a completely conflict-free airspace. The goal of this stage is to suggest partial solutions which result in a better overall airspace in terms of the criticality and difficulty of the induced conflicts. If the convergence stage fails to find any solutions which result in an improved airspace, the system will call the controller for help. Under these circumstances the system will provide the controller with all of the information known about why strategies have failed.

Ranking the Solutions

Solutions that are found by any of the three main stages are ranked and ordered according to suitability criteria and then presented to the controller. Some examples of the criteria used are, the expected deviation in flying time (as a result of the resolution procedure), the number of times an aircraft has been maneuvered previously in order to resolve conflicts, and the ease of maneuvering the particular plane (weight class).

Conflict Prediction

An integral part of the conflict resolution system is the conflict prediction mechanism. The prediction mechanism provides the initial conflict information to the resolution algorithm and is also used to test each strategy proposed by the resolution algorithm. The computation of accurate resolution advice is dependent on accurate conflict prediction information.

Unlike other automated conflict prediction mechanisms, which typically test aircraft separation using constant or variable sized circles drawn around each aircraft, the conflict prediction mechanism in CORES uses a myriad of ATC procedural separation rules. This requires the prediction mechanism to fully analyze the conflict situation in order to recognize which rules to apply (e.g., crossing track versus same direction rules).

2.2.2 Initial Resolution Stage

The Initial Resolution Stage consists of a number of sequential steps as shown in Figure 7. The steps are as follows:

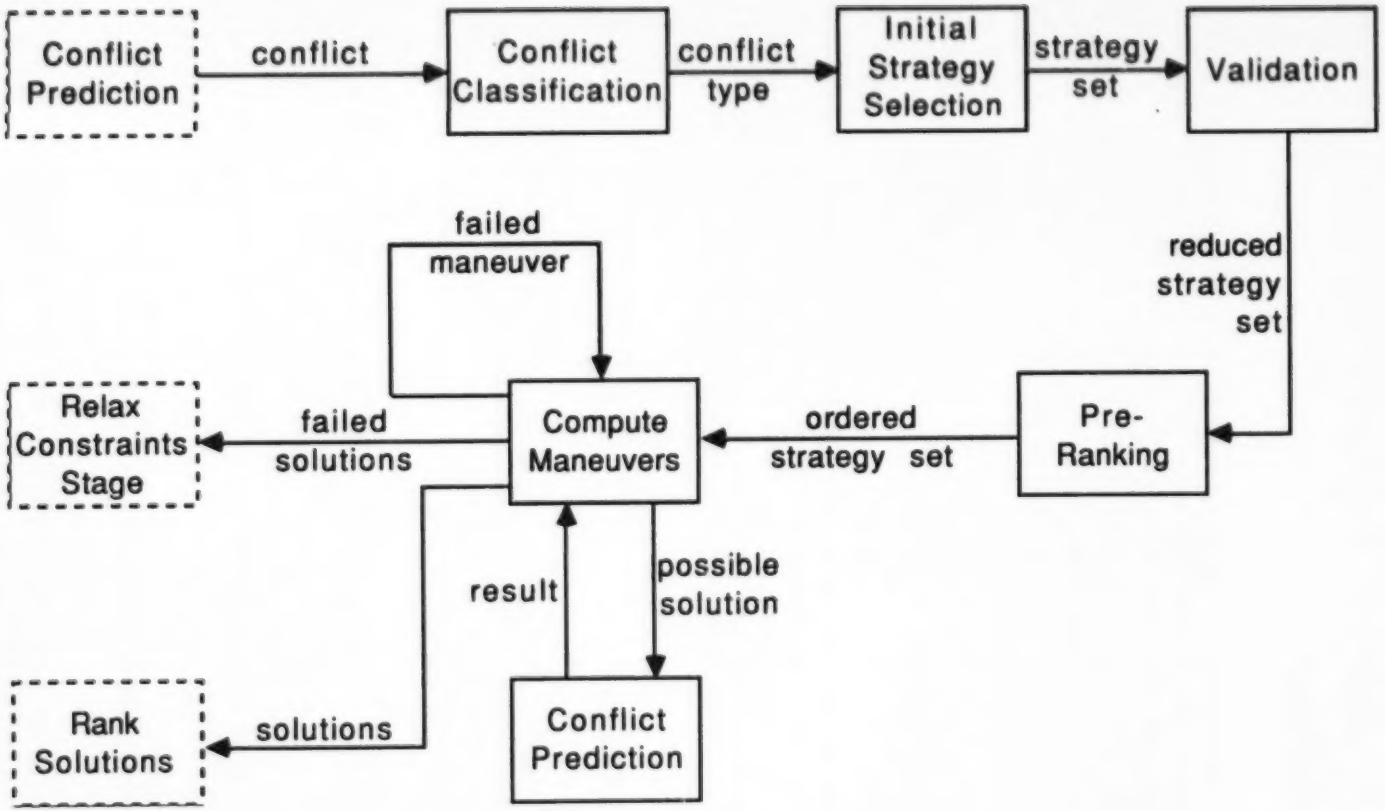


Figure 7: Initial Resolution Stage

1) **Classify conflict.** The algorithm classifies the conflict type based on the attitudes of the planes at the point of conflict. More classification types are used than in actual air traffic control. For example, head-on and overtaking conflict types have been added as special cases of same direction and opposite direction types respectively. Also, conflict classifications include whether the aircraft are in level flight, descent, or are climbing.

2) **Initial Strategy Selection.** Using the conflict type determined in step 1, an initial set of resolution strategies is selected. This set of strategies represents all possible strategies that may work for the conflict. At this step, all unrealistic strategies are eliminated from consideration. For example, in the case of a plane overtaking another, it would be illogical to decrease the speed of the leading plane to try to avoid the conflict.

3) **Validation.** Next, a validation check is performed on each strategy. This is based on aircraft performance characteristics, boundaries of the airspace and flight plan specific requirements. All those strategies which fail at this stage are excluded from future consideration.

4) **Pre-rank solutions.** All of the valid strategies obtained from step 3 are ranked according to their desirability. For example, in general a short delay is preferable to a long hold. This ranking is only approximate, and may not be the same as the final ranking presented to the controller. The strategies are preranked so that what are generally better strategies are tested first. This is important when the algorithm has been instructed to compute only a certain number of valid solutions (rather than find all possible solutions).

5) **Compute maneuvers and test.** For each strategy that passes the validation check, the exact maneuvers that are required to execute the strategy are computed. The maneuvers are then tested by the conflict prediction function to see if any induced conflicts occur as a result of executing the maneuvers. If the maneuvers computed for a particular strategy fail, a variation of the maneuvers may still work (e.g., trying to execute the maneuvers at an earlier time). If a variation is possible it is computed and tested. Any solutions found are passed to the Rank Solutions stage before being presented to the controller. Information on strategies that fail is passed to the relax constraints stage where further solutions may be computed.

2.2.3 Relax Constraints Stage

The Relax Constraints stage uses information about solutions that have failed from the compute maneuvers step of the Initial Resolution Stage. The relax constraints stage consists of a number of sequential steps similar to the Initial resolution stage as shown in Figure 8. The steps are as follows:

1) **Relaxed Strategy Selection.** Based on the solutions that failed from the Initial Resolution Stage, additional strategies are selected. The strategies are more extreme in nature (i.e., minimal perturbation constraints are relaxed). For example, strategies which route aircraft to wrong-way altitudes or more extreme rerouting of aircraft may be attempted.

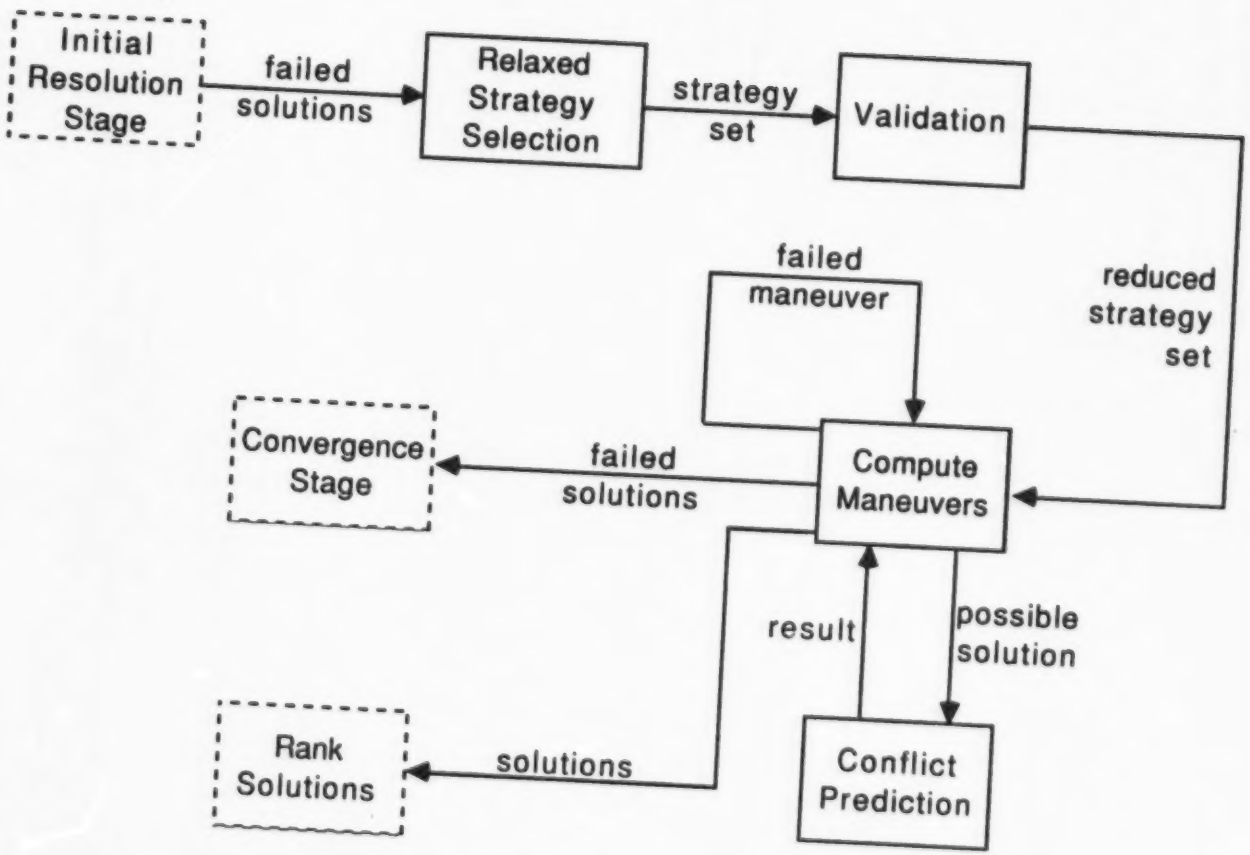


Figure 8: Relax Constraints Stage

2) **Compute maneuvers and test.** As in the Initial resolution stage, the exact maneuvers that are required to execute the strategy are computed and tested. Any successful solutions found are passed to the Rank Solutions stage before being presented to the controller. Information on strategies that fail is passed to the Convergence stage where partial solutions may be computed if necessary.

2.2.4 Convergence Stage

As shown in Figure 9, each of the strategies that have failed the Initial Resolution and Relax Constraints stages, because they induced conflicts, are evaluated against the original conflict situation (the one that is being solved). If the result of using one of these strategies is a "better" airspace, then the strategy is presented to the controller as a partial solution. The strategies are evaluated using two criticality criteria, time and complexity, as explained below. If a partial solution is selected for execution by the controller, the system will automatically begin trying to solve the remaining induced conflicts at an appropriate time.

1) **Criticality Test (time).** If the nearest induced conflict is later than a specified time (e.g., 20 minutes away), it is considered non-critical. Conflicts beyond this time may not require immediate action (although they may require immediate planning) and due to the dynamic nature of the airspace and errors in predicting flight paths, they may not even occur.

2) **Criticality Test (complexity).** A strategy may induce one or more conflicts. We generalize here by saying that the fewer conflicts the execution of a strategy produces, the less complex these induced conflicts will be to resolve (although this is not necessarily the case). This criterion is not as important as the time-criticality test.

3) **Evaluate Against Original Conflict.** The original conflict is also evaluated under the above criteria. A strategy is considered a partial solution if it compares favourably against the original conflict evaluation. For example, if the original conflict is critical (e.g., 5 minutes away) then it is acceptable to induce a non-critical conflict (e.g., 30 minutes away).

If a partial solution is selected for execution by the controller, then the system will automatically evaluate the situation again at a given time. If the selected solution induces a critical conflict, the system will evaluate the situation immediately. If the selected solution induces a non-critical conflict, the system will not evaluate the situation again until right before it becomes critical.

4) **Call for help.** The final step in the algorithm only occurs if all strategies result in an airspace which compares unfavourably with the original conflict situation. This would only occur if the original conflict was critical and all strategies induced a more critical conflict, or if the original solution was non-critical and all strategies induced a critical conflict. All information on why strategies have failed is available to the controller.

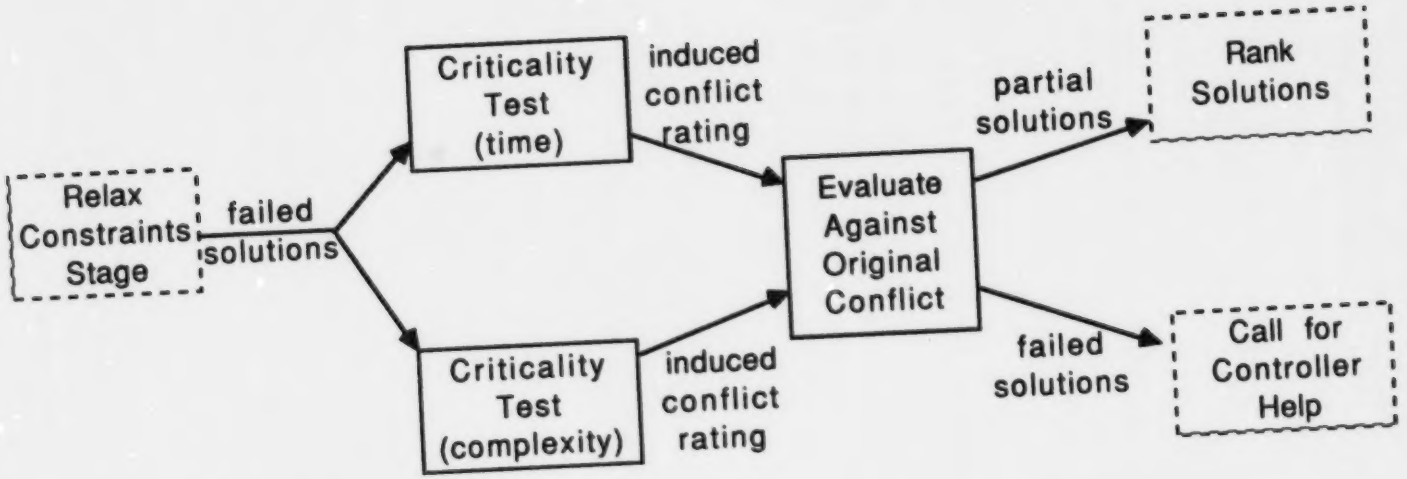


Figure 9: Convergence Stage

2.3 Assumptions

Since the purpose of CORES is to analyze the feasibility of the resolution algorithm, every feature of routine ATC is not simulated. As a result, the following assumptions and simplifications have been made:

1) Changes in wind and delays are all represented in the form of events that affect the state of the airspace. As no separate wind model is being used, the effects of known winds can be simulated by entering different ground speeds for different legs in the flight plan. Changing winds, simulated by events, also alter the ground speed.

2) In order to simplify the demonstration, planes are not allowed to come off-course or change altitudes as a result of an event. They may be either ahead or behind schedule or predicted to become off-schedule.

3) Whenever an illegal event arises during the course of a flight (for example, an event that causes an immediate loss of separation), then the illegal event is not executed. A message is displayed on the screen and recorded in the history log.

4) It is assumed that all planes entering the airspace have been cleared for entry (i.e., hand-offs have been cleared). As instantaneous conflicts are not appropriate, if during the demonstration an aircraft is defined to enter the airspace at a time which would cause this to occur, then the entry time for the aircraft is automatically redefined to a later time.

5) Aircraft may have flight plans which dictate they become active in the middle of the airspace; these must all begin at a fix within the airspace. All flight paths end at defined exit points at the boundary of the airspace (no flight path may terminate in the middle of the airspace).

6) Despite the fact that the flight plans are stored prior to the start of the demonstration, all planes are treated as unexpected. No provisions are made for the planes not yet in the airspace. This is consistent with the assumption for hand-offs detailed above.

7) Restricted airspaces are assumed to be generic, rather than there being various types. They correspond to such things as severe weather formations and military zones. Each of these airspaces is characterized by a perimeter that extends from the ground to infinity. Furthermore, restricted airspaces cannot move and are always active.

8) For the sake of simplicity, the simulated airspace is assumed to be flat. No provisions are made for the curvature of the earth close to the poles.

9) Since this is a demonstration rather than a real-time application of the resolution algorithm, the clock is stopped during the resolution search and selection phases of the demonstration. This is done so that the dependability and correctness of the system can be measured without affecting the system's ability to handle potential conflicts. Because the review of resolution logic would not normally be performed in a real-time situation, the clock need not be running.

10) The demonstration clock is not a real-time clock. The rate at which the clock ticks may vary with the load on the demonstration driver. However, the timings calculated during the resolution process are correct elapsed times.

11) The demonstration does not re-create any special cases of air traffic control that are not necessary to show the algorithm. In particular, no special air traffic procedures are required in the simulated airspace for such things as fuel dumping, hijackings, bomb threats, due regard flights and emergency flights. Furthermore, the system does not simulate such things as requests for other flight levels, requests for language communications, and other controller functions that are not essential to the primary task of proving the resolution algorithm.

12) There are no provisions for controllers to enter their own resolution procedures. However, a feature is provided to indicate where controllers may enter other procedures if CORES was an operational system.

13) Once a potential conflict is detected by the system and one or more resolution procedures are forwarded, a resolution procedure must be chosen. The potential conflict cannot be ignored. (Note: If the system fails to find any solutions, and 'calls for help', the conflict can then be ignored.)

14) No special treatment of inertial navigation system (INS) equipped aircraft is assumed, although planes may have flight plans not tied to NAVAIDs. The algorithm assumes that all planes can fly on all defined tracks and airways, and it is up to the controller to select the appropriate resolution provided by CORES.

15) Since the primary goal of this project is to test the conflict resolution algorithm, the conflict prediction mechanism works with a strict subset of separation criteria articulated in both the Air Traffic Services Administration and Management Manual (ATSAMM) and the Manual of Operations (MANOPS). See Appendix E in the CORES User's Guide [1] for a description of the separation criteria used.

2.4 Miscellaneous Information

This section lists miscellaneous facts and figures on the construction of CORES. Caution should be exercised when comparing these figures to similar figures for other systems. CORES was developed in Lisp using the KEE expert system builder. The size of a similar

application written in another language such as C would be significantly larger. Also, all of the following figures are CORES application specific (i.e., they do not include the size of any KEE routines).

There are approximately 20,000 lines of fully commented LISP code. This does not include knowledge base code.

There are an additional 8,400 lines of knowledge base code. Knowledge base code is CORES specific but is automatically generated by KEE (and as such it is not commented).

The percentage of knowledge base code devoted to the conflict resolution algorithm is 10.7%. The remaining 89.3% is dedicated to the demonstration system.

The percentage of application LISP code dedicated to the conflict resolution algorithm (including the conflict prediction function) is 43%. The remaining 57% is dedicated to the demonstration system.

Out of the combined LISP code for the conflict resolution algorithm and the conflict prediction function, the percentage devoted to the conflict resolution function is 65.1%; the percentage devoted to the conflict prediction function is 34.9%

3. EVALUATION

This section presents a critical evaluation of CORES. The evaluation is subdivided into a discussion of the conflict resolution algorithm and of the demonstration system. The evaluation of the algorithm is presented first because of its importance. The conflict resolution algorithm was evaluated using a set of scenarios that had previously been reviewed by the project's technical advisors. The evaluation of the demonstration system is primarily based on comments from the many people who received demonstrations of the system while it was being developed. The interpretations of both the test results and of the comments are based on our own subjective opinions.

The performance and capabilities of both the conflict resolution algorithm and the rest of the demonstration system are affected by limitations of the KEE expert system builder platform that was used to develop CORES. These limitations are discussed first.

3.1 KEE Performance Limitations

CORES was developed using the 386 version of the KEE expert system builder from Intellicorp. This version of 386-KEE (the August 29, 1988 release of version 3.1) fell below the expectations and requirements of the project. The reasons for this will not be discussed here; instead, a list of the KEE-related problems and limitations which severely affect the performance of the system are presented. Note that this should not be read as an evaluation of 386-KEE nor as a complete list of the problems that were experienced with the product during development of CORES.

1) Stop and copy garbage collection.

   The version of Lisp on which 386-KEE is based uses "stop-and-copy" garbage collection. Rather than continuously trying to reclaim memory, the system will halt when memory usage reaches a certain level and reclaim as much as possible all at one time. This reclamation typically takes between one and three minutes while running CORES. Having the system halt for this amount of time severely detracts from the impact of the demonstration. This also makes obtaining accurate performance timing results for the resolution algorithm difficult.

2) Memory management.

   Due to a problem in the Unix virtual memory management (Unix comes bundled with 386-KEE), the system may halt for extended periods of time (e.g., 3 to 5 minutes) depending on the process size. In order to minimize this problem, CORES was developed under the constraint that the process size must be kept to a minimum.

3) No colour or high resolution graphics.

There is no support for colour graphics within KEE. There is also a problem with KEE which prevents CORES from running at a higher resolution than EGA.

4) Poor graphics performance.

The graphics routines in KEE provide slow performance and in some cases seem to use an unreasonable amount of memory. It takes an excessive amount of time to display aircraft, menus, warning messages, etc. in CORES.

5) No multi-tasking.

The LISP used in 386-KEE is not multi-tasking. This means that when one process is running and another one starts, the second process must finish before the first will resume. This meant that the conflict resolution algorithm described in Section 2.2 had to be implemented in a strictly sequential fashion (e.g., the initial resolution stage must finish completely before the relax constraints stage can start).

6) LISP not suited to number crunching.

CORES is entirely written using LISP or KEE routines. Portions of CORES are well suited to symbolic processing (which LISP is good at) but other portions are strictly concerned with "number-crunching" (which LISP is not best suited for). Although KEE provides a mechanism for calling foreign language routines, the overhead of converting the common data structures so that they could be operated on by both a foreign language and LISP was excessive.

7) KEE not designed for delivering applications.

386-KEE provides a large number of expert system development tools. However, most of these tools are unnecessary when running the fully developed version of CORES. There is no way of stripping-out or excluding portions of the development tools. This extra overhead may contribute to the poor run-time performance of 386-KEE.

3.2 Algorithm Tests

In order to test all three stages of the algorithm (as described in Section 2.2), scenarios were specifically developed for each stage. The complete details of these scenarios can be found in the CORES Test Document [3]. This section summarizes the results of running these scenarios and identifies strengths and limitations of the algorithm.

3.2.1 Advice Produced by the Algorithm

Due to time constraints of both the technical advisors and of the CORES development team, only a limited number of scenarios were developed for each algorithm stage. There were three simple scenarios to test the initial resolution stage, two scenarios for the relax constraints stage, two scenarios for the convergence stage, and one scenario to test the call for help function.

Overall, the resolution advice presented by the system seems reasonable under the constraints of the system. All of the scenarios were successfully executed as specified. The main criticism of the resolution portion of the system is not the type of advice produced but the time it takes to produce it.

In demonstrations, the convergence stage of the algorithm successfully solved the given situations. Where all strategies induced further conflicts, the 'best' strategies were chosen, and at a suitable future time (e.g., 20 min. before the induced conflict is to occur), the induced conflict was evaluated and solved. Although different solving techniques may be employed (e.g., the controller may have already solved the induced conflict before issuing any instructions to solve the original conflict), the final result was acceptable to those who saw this demonstrated.

The call for help stage of the algorithm was rarely required, and seemed to be only in unrealistic situations. For example, a stack of planes is flying on the same track at all altitudes, a plane pops up on the same track flying in the opposite direction, and the conflict is predicted to occur in 5 min.

3.2.2 Algorithm Performance

The amount of time taken to solve a given conflict varied greatly (see table below), from 25 seconds to 4.5 minutes (this is the elapsed time from the detection of the conflict, to the time the solutions are displayed). The timing for the resolution algorithm itself also varied from 16 seconds to 1 minute (this did not include the time taken predicting induced conflicts for all the possible maneuvers). This variance is due to a number of factors: the number of garbage collections which occur, the type of conflict (a restricted airspace conflict will discard many strategies at the very beginning, where a crossing track conflict will continue testing all strategies), at what stage in the algorithm solutions were found (initial resolution, relax constraints, or convergence), and bugs in the underlying operating system (see 3.1;2 Memory Management).

Resolution timing table

	Total time	Resolution time	Conflict Detection time
Scenario 1	0:25	0:18	0:00
Scenario 2	0:40	0:23	0:09
Scenario 3	0:56	0:22	0:09
Scenario 4	2:06	0:32	1:24
Scenario 5	3:50	0:46	2:55
Scenario 6	3:39	0:47	2:41
	4:24	1:02	3:12
Scenario 7	0:56	0:16	0:14
	1:01	0:22	0:31
Scenario 8	1:43	0:52	0:43
Average	1:53	0:34	1:12

The resolution timing table represents the elapsed times calculated during final testing of the CORES system. The scenarios are described in the CORES System Testing Document [3]. The total time is the elapsed time from conflict detection to the presentation of solutions. The resolution time is the solving time (not including display or conflict detection). The conflict detection time is the time spent testing the detailed maneuvers for induced conflicts. Scenarios with two sets of times represent convergence stage scenarios; the second set of times is for solving the induced conflict.

The number and length of garbage collections, and the bug in the operating system made it very difficult to obtain accurate and repeatable timing results; running the same scenario twice could result in different performance timings. In an attempt to minimize this, an optional parameter was added which allows the user to specify that a garbage collection always be executed immediately before a conflict is solved.

The most time consuming portion of the resolution algorithm (not including the conflict prediction process), is computing the detailed maneuvers for each strategy. While each computation is relatively quick, this process can be called anywhere from one to sixty or more times while solving a conflict; each strategy is applicable to each plane, and many have a number of different times at which the maneuver can begin.

Due to the fact that many different timings are tested for some strategies, and this is very time consuming, the method of choosing the timing for each attempt, should be more intelligent. Currently, most strategies that have more than one timing to test, simply compute the maneuver to begin ten minutes before the last attempt. For example, if we first attempt to climb a plane to FL310 at 1400h, and this results in an induced conflict, we will try climbing at 1350h, 1340h, etc., until we run out of different times to try, or until we find one that does not induce conflicts. Determining whether or not to continue with different timings should also be more intelligent; in the previous example, if the induced conflict was with a plane

following the same track and direction at FL310, then climbing at 1350h will obviously not work.

Due to the stage in the algorithm at which strategies are discarded, some simple scenarios may take longer than more difficult ones. For example, a conflict which has ten solutions can take longer than a conflict which has only one possible solution. This can occur if, while solving the one solution conflict, most of the strategies were discarded before computing the maneuvers and testing for induced conflicts, which is the most time consuming portion of the resolution algorithm. The ten solution conflict had to compute maneuvers and test for induced conflicts at least ten times.

3.2.3 Conflict Prediction Performance

When the project to develop CORES began, we completely underestimated the complexity and effort required to develop the conflict prediction mechanism. Although a simpler prediction algorithm would have been sufficient to demonstrate the functionality of the resolution algorithm, it would have detracted from a controller's perception of efficient airspace utilization. Interestingly, the conflict prediction mechanism is a performance bottle-neck. On average when solving a conflict, 63% of the time is spent testing potential resolution procedures, with most of the remainder spent selecting and computing the resolution procedures. However, to our knowledge no other automated conflict prediction mechanism intelligently applies as many ATC rules and procedures.

Other performance penalties came from integrating a graphics display system and a simplified simulator with the conflict prediction and resolution systems on the same processor.

3.2.4 Algorithm Design Issues

The following section identifies some of the major issues that occurred and were resolved during the course of designing and implementing the resolution algorithm.

The original design of the algorithm was proposed using rules as the main method of control. However, in the final analysis, rules were used for the initial strategy selection, and the validation; this is about fifteen rules in all. As it turned out, the resolution algorithm has a specific ordering; and one of the main features of a rule-based system, is that it is difficult (and not usually necessary) to control the order in which rules are tested.

During the course of this project, the possibility was mentioned of completely incorporating the MANOPS into a system such as this. In the resolution of a conflict, the actual procedure applied from the MANOPS could be displayed. However, due to the complexity, and the sheer volume of rules and procedures, this would be a nearly impossible task.

Once the design of the resolution algorithm was completed, the most difficult task turned out to be implementing the processes to compute the maneuvers. Each different strategy required a different process to compute the detailed maneuvers, and computing more extreme strategies also affected these procedures. There were also many assumptions to be made in the determination of the details, such as the rate of climb that should be used for a plane, or how to define a holding pattern, and what size it should be. The details of the maneuvers also had to take into account the separation rules and assumptions used during the conflict prediction; for example, geographical separation is not recognized by the conflict prediction function, therefore the maneuvers cannot be computed based on the assumption of geographical separation.

The original algorithm specification called for 2-maneuver strategies, e.g., climb and hold. This was not implemented, and in retrospect, we would not suggest that it be tried in the future. Instead, a limited look-ahead should be used, in which induced conflicts from the climb (in the above example) are evaluated first, and possibly then solved using a hold, or some other method. The implementation of 2-maneuver strategies is more like a brute force approach, where a limited look-ahead would reduce the number of second maneuvers which would have to be tested.

The design of the ranking procedure also presented a problem, as it was difficult to obtain any concrete methods that controllers use to select one solution over another. Also, some criteria that were mentioned (such as point of departure and destination), are not known by CORES.

3.2.5 Unresolved Man-Machine Interface Issues

Although the man-machine interface was not a major concern in this project, there remain a number of issues that were raised during the course of the project, and never resolved. These will be limited to those directly related to the display and presentation of resolution advice.

The first point involves the presentation of the ranking for each solution. Due to the fact that the ranking is based on incomplete knowledge (the controller will probably have additional information, e.g., fuel load, pilot preferences), and controllers having their own biases, it has been suggested that the ranking not be displayed at all. The calculation, and the order of the solutions may remain, but not the actual values.

Along the same lines, is the question of how many solutions should be displayed to the controller; one, three, all of them, etc. The presentation of only one solution presents the obvious problem that it may not be the best one (due to the incomplete knowledge in the ranking procedure). However, presenting all the solutions can make the task more difficult than it should be. For example, usually the simplest conflicts result in the most possible solutions, and an overabundance of possibilities for the controller to evaluate. Possibly there should be a different number of solutions presented, depending on the complexity or uncertainty of the problem.

3.3 Demonstration System

This section describes general weaknesses of the demonstration system. This evaluation is based on comments from a variety of people who received demonstrations of the system, both while it was being developed and after the system was completed.

i) The graphics capability available for KEE on the 386 was inadequate for the high degree of graphics required. Low resolution, and the inability to plot in color are just some of the drawbacks attributable to this package. In particular, the absence of high definition graphics made the display of several planes within the same area on the screen rather cluttered. When planes overlap, their leaders were extremely difficult to differentiate. Both high resolution and color would rectify this.

ii) The overall speed of KEE on the 386 was inadequate for a system the size of CORES. While the system provided adequate facilities for producing menus and entering in data, etc., the time it took to switch between screens and menus was almost prohibitively slow.

iii) Several features of the interface are redundant or clumsy and need not have been designed that way. For example, entering the time into the data window of the SET CLOCK option required two actions, one to enter the data and another to alert the system to continue, when only one would do.

iv) A desirable feature would be to enable the user to enter a change to a flight plan or an event "on the fly". This would mean that execution of the demonstration could be interrupted intermittently in order for the user to modify existing flight plans.

v) Concurrency could be exploited by the system in order to increase performance. Specifically, solving could occur at the same time that planes are being moved. In that way, the user can still observe the airspace. Should a second conflict arise while the first conflict is being solved, then a second solving window would appear and the two conflicts would be solved concurrently.

4. CONCLUSIONS

The goals of the project were all achieved; CORES demonstrated the feasibility of an automated conflict resolution advisory system, and the feedback from the many people who received demonstrations has been positive. The problems identified were often directly attributable to the software/hardware platform being used, such as the speed of the system, or the graphics capabilities. There were also a number of recommendations for follow-on work which were suggested during the course of the project, some of which will comprise our next major research effort.

The feasibility of an operational conflict resolution advisory system has been demonstrated. Unlike other prototypes which are incomplete or limited to restrictive types of scenarios, CORES provides a vehicle for testing the effectiveness of an automated advisory function under a wide range of circumstances (including traffic patterns which are unrealistic in today's environment). Issues for future study include, how to best integrate this type of system into either current or future operations and how to best present resolution advice in an effective manner.

5. RECOMMENDATIONS

It has been proposed that the current CORES system be extended for use in general flight planning applications. It could be applied to both the preplanning of flights and to strategic planning for dynamic in-flight adjustments. The advantage of this system is that it could predict controller workloads not only in terms of the number of aircraft per sector but also in terms of the difficulty of possible conflict situations that may arise.

As an extension, the system simulation capabilities could be adapted to evaluate new route structures or changes in procedures in terms of controller workload.

There are difficulties in adapting the CORES system directly to a radar environment. The main reason for these difficulties is the lack of facilities for an automated system to accurately predict aircraft flight paths between approximately 3 and 20 minutes ahead. Currently, for the non-radar environment, CORES predicts flight paths using flight plan information and pilot position reports. In a radar environment, flight paths can be predicted up to about three minutes ahead using only radar returns, and flight plans are only suitable for deriving very long term intent. Neither of these sources are suitable for predicting accurate aircraft trajectories between 3 and 20 minutes ahead. Flight plans in a radar environment are missing critical information. For example, small vectoring adjustments to traffic may be planned by a controller, but are not recorded anywhere for an automated system to access. The introduction of Automatic Dependent Surveillance (ADS) will dramatically change this.

However, before ADS is common place there is a need to investigate automated tactical/emergency conflict resolution advisory for controllers. Unlike the strategic system used in CORES, the overriding constraint of a tactical system is immediate, accurate response. The advice presented must also be in a form which is instantly understandable and useable to a controller. Meeting these requirements will be challenging. Developing a prototype tactical (radar-based) conflict resolution advisory system will be our next major research and development effort.

Further development of CORES can assist in the future design of an operational conflict resolution system. This could include such work as:

- Investigating how the conflict prediction and resolution functions can keep better track of relevant information constantly, rather than discarding past information and (often) recalculating the same information later.

- Expanding the possible maneuvers to include other types of separation used by controllers, such as omni-track, or geographical separation.

- Calculating the different times at which to begin maneuvers should be more intelligent, and evaluate the induced conflicts resulting from the previous timing of the maneuver. This is especially important, as the detailed maneuver computations are time consuming.

- Due to the limited time frame in which to produce testing scenarios, some of CORES' features were not adequately tested under all circumstances; it would be helpful to have

more statistics regarding how often and why the system fails to find a solution, or what types of conflicts are being induced during the convergence stage.

- The ranking of solutions should also be further investigated, and more information should be included in the calculation.

- The dynamic modification of flight plans should be investigated to allow the user to alter scenarios while the demonstration is running, rather than having to exit the demonstration, and enter the update portion of CORES.

- The current stub that is in the place of the IGNORE/ENTER SOLUTION option on the solutions screen, should be replaced to allow the user this option, rather than simply stating that it would be here in an operational system.

- Further work would also precipitate the move to a new, and faster, hardware platform and version of KEE; it is infeasible to abandon KEE simply to accommodate additions to the current system.

- A detailed man-machine interface study should be done to determine effective methods of presenting conflict resolution advice to controllers. The study should also examine methods of entering controllers' own resolution strategies.

6. REFERENCES

This section lists all documents explicitly referenced in this report. References are listed in their order of occurrence in the body of this report.

1. CompEngServ Ltd., CORES: Air Traffic Conflict Resolution System User's Guide, Rev. 1.00F, Ottawa, July 1989.

2. CompEngServ Ltd., CORES: Air Traffic Conflict Resolution System Software Design Document, Volumes 1 through 5, Rev. 1.00F, Ottawa, July 1989.

3. CompEngServ Ltd., CORES: Air Traffic Conflict Resolution System Testing Document, Rev. 1.00F, Ottawa, July 1989.

4. Beaton, J.D. An Expert System For Conflict Avoidance Algorithm Development - Prototype Design, Rev. 1.00F, CompEngServ Ltd., Ottawa, March 1986.

5. Bowen, B.A., Beaton, J.D., Standen, N.M. A Conflict Resolution Algorithm for Sparse Air Spaces using a Hybrid Knowledge-Based System, Canadian Aeronautical and Space Institute, Sixth Canadian Symposium on Navigation, Vancouver, B.C., May 1986.

6. Bowen, B.A., Beaton, J.D., Standen, N.M. Feasibility Demonstrated for Automatic Conflict Resolution, ICAO Bulletin, September 1986, pp. 25-27.

7. Bowen, D.G., Bowen B.A. Specifications For a Conflict Avoidance Expert System in Dense Airspace, CompEngServ Ltd., Ottawa, January 1987, TP 8224E.

8. Bowen, B.A. An Expert System For Aircraft Conflict Resolution in Dense Airspaces, in Advisory Group for Aerospace Research and Development, North Atlantic Treaty Organization, August 1987, pp. 4-1 to 4-7.

7. BIBLIOGRAPHY

This section lists all documents which are not explicitly referenced in this report but which provide related, supplementary material. A brief description of contents follows each reference.

Smith, I.G., Bowen D.G. Automated Conflict Resolution Advisory: A Key Function for Airspace Management in 2020, Submitted to Airshow Canada Symposium on the Future of Air Transportation, Vancouver, August 8 to 10, 1989.
[*Describes how automated conflict resolution and prediction fit into future ATC plans (ICAO, TC, FAA, etc.). Gives a brief overview of the algorithm. Concludes with a discussion of pertinent issues including how improved surveillance (i.e, ADS) will make automated conflict resolution possible.*]

Smith, I.G., Bowen D.G., Bowen, B.A. CORES: A Knowledge-Based System For Aircraft Conflict Resolution In Non-Radar Airspace, Proceedings of the Canadian Conference on Electrical and Computer Engineering, Vancouver, November 3-4, 1988, pp. ??-??.
[*An early paper which describes the initial design of CORES. This design was later modified slightly, as presented in this report.*]

Smith, I.G., Crabb, C.A., Attisha, M.R., Ram, G.R. A Unique Approach To Aircraft Conflict Resolution Using Artificial Intelligence Techniques, Submitted to Canadian Conference on Electrical and Computer Engineering, Advanced Systems for Air Space Management Group, Montreal, September 17 to 21, 1989.
[*Presents a nigh level overview of the CORES resolution algorithm and an example scenario. Concludes with a brief summary of the results of the project and the direction for the future.*]